In the United States Patent and Trademark Office

| | | |
|---|---|---|
| In re the application of: Palliyil | | |
| Filed: 12/12/2003 | Group Art Unit: | 2139 |
| For: Apparatus, methods and computer programs for identifying matching resources within a data processing network | Examiner: | James R. Turchen |
| Appl. No.: 10/735,509 | | |
| Appellant's Docket: JP920030270US1 | | |

## SUBSTITUTE SUMMARY OF CLAIMED SUBJECT MATTER FOR APPEAL BRIEF

This is responsive to a Notification of Non-Compliant Appeal Brief of June 26, 2009, which indicates that the Summary of Claimed Subject Matter fails to refer each independent claim to the specification by page and line number. (The specification was referred to by paragraph number instead, i.e., paragraph numbers of the published application.)

Please enter this Substitute Summary of Claimed Subject Matter section submitted herein .

SUBSTITUTE SUMMARY OF CLAIMED SUBJECT MATTER

The present application provides context for what is claimed, as follows:

Page 10, lines 9-18: In a local area network environment, it is common for each personal computer 70 to have a similar set of installed computer programs, and for some of the data files stored within the LAN to be replicated across several computers in the network. Therefore, periodic executions of the antivirus software typically involve scanning identical data files and executable files on many different computers. The periodic virus scans involve scanning newly created and newly installed files, but also repeating virus scans of files which were already in existence when the last virus scan was performed. The pre-existing files may not have changed since the last scan, but repeated scanning of pre-existing files has previously been considered essential for protection because timestamps on files cannot be relied on as evidence that the files have not changed.

Page 10, lines 20-27: The inventors of the present invention have identified these issues as problems requiring a solution. Embodiments of the invention described below use a comparison of hash values computed from the bit patterns representing stored files to identify which files have changed since the last virus scan. The embodiment avoids full virus scanning of files which have not changed since the last scan. Another feature, or alternative embodiment, of the invention also uses a comparison of hash values to identify replicas of files to avoid repetitious virus scanning of multiple replicas. Further embodiments are described thereafter.

. . .

Page 30, lines 13-24: A further embodiment of the invention uses statistical observation of the pattern of creation of new hashes to identify sudden changes within a network. For example, if newly computed hash values are compared with stored hash values and a large number of copies of a specific hash value $MD_1$ can be seen to have changed, this implies that the corresponding copies of the resource represented by hash value $MD_1$ have also changed. This could mean that a group of users are upgrading from one file version to another (for example if $MD_1$ consistently changes to $MD_2$) or that a virus is spreading through the system. The latter is most likely if a large number of copies of $MD_1$ have remained unchanged for a long period and are then suddenly replaced by a large number of different hash values--indicating the probable spread of a polymorphic virus. The comparison of hash values can be used once again to determine which resources require a virus scan and which do not.

Claim 24

Claim 24 describes a method for computing first hash values derived from and representing a plurality of replicas of a resource, wherein the replicas are stored on respective data processing systems within a network. The claim has steps as follows:

Step: storing the computed first hash values;

Step: computing current hash values for the replicas of the resource;

Step: comparing the current and first hash values in order to identify whether all the hash values match, wherein nonmatching first and current hash values for a respective one of the replicas indicates the respective one of the replica has changed since the computing of the first hash value;

Step: detecting that a vulnerability exists responsive to the hash value comparison indicating more than a predetermined number of changed replicas of the resource, and that no vulnerability exists responsive to the hash value comparison indicating less than or equal to the predetermined number of changed replicas, wherein the predetermined number is at least one; and

Step: presenting a message for a user indicating a vulnerability, wherein the presenting is responsive to the predetermined number being exceeded.

The specification of the present application provides an exemplary embodiment of the invention and describes the method of claim 24 in terms of that embodiment. Specifically, regarding support for claim 24, see application as filed, page 13, lines 3-12, Fig. 2 (computing 200 first hash values derived from and representing a plurality of replicas of a resource, wherein the replicas are stored 220 on respective data processing systems within a network); page 13, lines 3-12, Fig 2 (storing 220 the computed first hash values); page 14, lines 1- 13, Fig. 3 (computing 300 current hash values for the replicas of the resource); page 14, lines 1-13, Fig. 3 (comparing 310 the current and first hash values in order to identify whether all the hash values match 320); page 12, lines 9-19, i.e., "If the file has been modified, a hash value computed after the change will differ from a hash value computed before the change . . ." (wherein nonmatching first and current hash values for a respective one of the replicas indicates the respective one of the replica has changed since the computing of the first hash value); page 29, line 23 - page 30, line 1, page 30, lines 13-24, Fig. 8 (detecting that a

vulnerability exists responsive to the hash value comparison 430 indicating more than a predetermined number of changed replicas of the resource, and that no vulnerability exists responsive to the hash value comparison 430 indicating less than or equal to the predetermined number of changed replicas, wherein the predetermined number is at least one); page 29, line 23 - page 30, line 1, Fig. 8 (presenting a message to a user indicating a vulnerability); and page 30, lines 13-24 (wherein the presenting is responsive to the predetermined number being exceeded).

Claim 31

Claim 31 describes an apparatus that includes a processor and a storage device connected to the processor. The processor is operative to execute instructions of the program to implement a method. The claim has steps as follows:

Step: computing first hash values derived from and representing a plurality of replicas of a resource, wherein the replicas are stored on respective data processing systems within a network;

Step: storing the computed first hash values;

Step: computing current hash values for the replicas of the resource;

Step: comparing the current and first hash values in order to identify whether all the hash values match, wherein nonmatching first and current hash values for a respective one of the replicas indicates the respective one of the replica has changed since the computing of the first hash value;

Step: detecting that a vulnerability exists responsive to the hash value comparison indicating more than a predetermined number of changed replicas of the resource, and that no vulnerability exists responsive to the hash value comparison indicating less than or equal to the predetermined number of changed replicas, wherein the predetermined number is at least one; and

Step: presenting a message for a user indicating a vulnerability, wherein the presenting is responsive to the predetermined number being exceeded.

The specification of the present application provides an exemplary embodiment of the invention and describes the apparatus of claim 31 in terms of that embodiment. Specifically,

regarding support for claim 31, see application as filed, page 9, lines 7-15 (a storage device connected to the processor, wherein the storage device has stored thereon a program, wherein the processor is operative to execute instructions of the program to implement a method); page 13, lines 3-12, Fig. 2 (computing 200 first hash values derived from and representing a plurality of replicas of a resource, wherein the replicas are stored 220 on respective data processing systems within a network); page 13, lines 3-12, Fig 2 (storing 220 the computed first hash values); page 14, lines 1- 13, Fig. 3 (computing 300 current hash values for the replicas of the resource); page 14, lines 1- 13, Fig. 3 (comparing 310 the current and first hash values in order to identify whether all the hash values match 320); page 12, lines 9-19, i.e., "If the file has been modified, a hash value computed after the change will differ from a hash value computed before the change . . ." (wherein nonmatching first and current hash values for a respective one of the replicas indicates the respective one of the replica has changed since the computing of the first hash value); page 29, line 23 - page 30, line 1 and page 30, lines 13-24, Fig. 8 (detecting that a vulnerability exists responsive to the hash value comparison 430 indicating more than a predetermined number of changed replicas of the resource, and that no vulnerability exists responsive to the hash value comparison 430 indicating less than or equal to the predetermined number of changed replicas, wherein the predetermined number is at least one); page 29, line 23 - page 30, line 1, Fig. 8 (presenting a message to a user indicating a vulnerability); and page 30, lines 13-24 (wherein the presenting is responsive to the predetermined number being exceeded).

Claim 38

Claim 38 describes a computer program product stored on a tangible, computer readable medium. The computer program product has instructions for execution by a computer system. The instructions, when executed by the computer system, cause the computer system to implement a method. The claim has steps as follows:

Step: computing first hash values derived from and representing a plurality of replicas of a resource, wherein the replicas are stored on respective data processing systems within a network;

Step: storing the computed first hash values;

Step: computing current hash values for the replicas of the resource;

Step: comparing the current and first hash values in order to identify whether all the hash values match, wherein nonmatching first and current hash values for a respective one of the replicas indicates the respective one of the replica has changed since the computing of the first hash value;

Step: detecting that a vulnerability exists responsive to the hash value comparison indicating more than a predetermined number of changed replicas of the resource, and that no vulnerability exists responsive to the hash value comparison indicating less than or equal to the predetermined number of changed replicas, wherein the predetermined number is at least one; and

Step: presenting a message for a user indicating a vulnerability, wherein the presenting is responsive to the predetermined number being exceeded.

The specification of the present application provides an exemplary embodiment of the invention and describes the computer program product of claim 38 in terms of that embodiment. Specifically, regarding support for claim 38, see application as filed, page 5, lines 7-18 and page 9, lines 7-15 (A computer program product, stored on a tangible, computer readable medium, said computer program product having instructions for execution by a computer system, wherein the instructions, when executed by the computer system, cause the computer system to implement a method); page 13, lines 3-12, Fig. 2 (computing 200 first hash values derived from and representing a plurality of replicas of a resource, wherein the replicas are stored 220 on respective data processing systems within a network); page 13, lines 3-12, Fig 2 (storing 220 the computed first hash values); page 14, lines 1- 13, Fig. 3 (computing 300 current hash values for the replicas of the resource); page 14, lines 1-13, Fig. 3 (comparing 310 the current and first hash values in order to identify whether all the hash values match 320); page 12, lines 9-19, i.e., "If the file has been modified, a hash value computed after the change will differ from a hash value computed before the change . . ." (wherein nonmatching first and current hash values for a respective one of the replicas indicates the respective one of the replica has changed since the computing of the first hash value); page 29, line 23 - page 30, line 1 and page 30, lines 13-24, Fig. 8 (detecting that a vulnerability exists responsive to the hash value comparison 430 indicating more than a

predetermined number of changed replicas of the resource, and that no vulnerability exists responsive to the hash value comparison 430 indicating less than or equal to the predetermined number of changed replicas, wherein the predetermined number is at least one); page 29, line 23 - page 30, line 1, Fig. 8 (presenting a message to a user indicating a vulnerability); and page 30, lines 13-24 (wherein the presenting is responsive to the predetermined number being exceeded).

REQUEST FOR ACTION

Please enter this Substitute Summary of Claimed Subject Matter section submitted herein for the corresponding section previously filed.

Respectfully submitted,

/anthony v.s. england/

Anthony V.S. England
Registration No. 35,129
Attorney of Record for
IBM Corporation
Telephone: 512-477-7165
a@aengland.com